sensors

MDPI

Article

Monocular Visual-Inertial Odometry with an Unbiased Linear System Model and Robust Feature Tracking Front-End

Xiaochen Qiu [1,2], Hai Zhang [1,2,*], Wenxing Fu [3], Chenxu Zhao [4] and Yanqiong Jin [1,2]

1 School of Automation Science and Electrical Engineering, Beihang University, No. 37 Xueyuan Road, Haidian District, Beijing 100191, China; qiuxiaochen@buaa.edu.cn (X.Q.); buaa_jinyq@buaa.edu.cn (Y.J.)
2 Science and Technology on Aircraft Control Laboratory, Beihang University, No. 37 Xueyuan Road, Haidian District, Beijing 100191, China
3 Science and Technology on Complex System Control and Intelligent Agent Cooperation Laboratory, No. 40 Yungangbeili, Fengtai District, Beijing 100074, China; fuwenxingCHN@163.com
4 Sino-French Engineer School, Beihang University, No. 37 Xueyuan Road, Haidian District, Beijing 100191, China; chenxu.zhao@buaa.edu.cn
* Correspondence: zhanghai@buaa.edu.cn; Tel.: +86-10-8233-9366

Received: 23 March 2019; Accepted: 22 April 2019; Published: 25 April 2019

Abstract: The research field of visual-inertial odometry has entered a mature stage in recent years. However, unneglectable problems still exist. Tradeoffs have to be made between high accuracy and low computation for users. In addition, notation confusion exists in quaternion descriptions of rotation; although not fatal, this may results in unnecessary difficulties in understanding for researchers. In this paper, we develop a visual-inertial odometry which gives consideration to both precision and computation. The proposed algorithm is a filter-based solution that utilizes the framework of the noted multi-state constraint Kalman filter. To dispel notation confusion, we deduced the error state transition equation from scratch, using the more cognitive Hamilton notation of quaternion. We further come up with a fully linear closed-form formulation that is readily implemented. As the filter-based back-end is vulnerable to feature matching outliers, a descriptor-assisted optical flow tracking front-end was developed to cope with the issue. This modification only requires negligible additional computation. In addition, an initialization procedure is implemented, which automatically selects static data to initialize the filter state. Evaluations of proposed methods were done on a public, real-world dataset, and comparisons were made with state-of-the-art solutions. The experimental results show that the proposed solution is comparable in precision and demonstrates higher computation efficiency compared to the state-of-the-art.

Keywords: visual inertial odometry; quaternion notation; closed-form state transition equation; robust feature tracking; real-time motion tracking; computation saving

1. Introduction

The fusion of monocular cameras and inertial measurement units (IMUs) is very popular recently, thanks to great improvements in the computation capacity of computers with low energy costs and low weight, and the increasing demand for accurate motion tracking or positioning in unmanned aerial vehicles (UAVs), augmented reality, and driverless cars. This fusion problem has been studied by brilliant scientists for years [1–4], and as a result, one can now build his own visual-inertial odometry (VIO) module simply with cheap sensors and open-source software [2,5–9].

Fusion frameworks are divided into two main branches: filter-based and optimization-based [10,11]. Optimization-based methods are so far widely recognized performing better in terms of precision [12] due to their iterating mechanism, which is essentially solving a noted bundle adjustment (BA) problem [13]. The BA problem was considered to be computational costly in earlier years, until the literature recognized and revealed its sparse structure [13,14] so as to develop real-time algorithms. Kümmerle et al. [7] modeled BA as a graph optimization problem. Kaess et al. [8] introduced a factor graph model to further illustrate the Bayesian nature of BA. Kaess et al. [8] also found that the incremental fact of sophiscated Hessian matrix in normal equation can be utilized for solving BA, thus speeding up the calculation further. With these profound insights, researchers also made efforts to overcome the inconsistency in fixed-lag fusion algorithms, which has the advantages of having bounded computation with less information loss and maintaining the sparse structure [15]. Several open-source libraries are available for building the back-end for algorithms of this branch, based on different mathematical descriptions listed above and providing convenient application program interfaces (APIs) [7–9,16]. Although enabling reduced computation by leveraging sparse matrix factorization, optimization-based VIO systems still need to be tailored sometimes in order to be deployed on a computation-limited platform. Sometimes, this leads to a downgraded performance [17].

Before the flowering of optimization-based methods, the solving of fusion problems was dominated by filtering [18]. The ordinary procedure is to include IMU pose and map point positions in the filter state and recursively propagate and update as IMU and camera measurements respectively arrive [10]. The accurate estimation of map point positions is the key to bring about an unbiased IMU pose updating. In traditional filter-based solutions, the filter state would invariably have a very large dimension since it always preserves a lot of map points, resulting in enhanced computation requirements [1]. The use of a multi-state constraint Kalman filter (MSCKF) was proposed as an effective and optimal filter-based solution that does not maintain map points in filter state [19]. By properly handling the camera measurement, MSCKF can achieve as competitive a performance as optimization-based algorithms and demands far less computation [17].

By correcting observability properties [19–21] and incorporating camera–IMU extrinsic parameters into the filter state [22], the performance of MSCKF was further improved. Many follow-up works emerged, including an open-source monocular implementation [23], expansion to stereo camera rig [24], and schemes using direct visual front-ends [25] or adding line features [26].

It should be emphasized that all members of the MSCKF family so far have been developed based on Shuster's notation of quaternion [27], whereas most of the community utilizes the traditional Hamilton notation, which results in unnecessary trouble in understanding for researchers [28].

Visual front-ends apparently play an important role in VIOs. There are typically of two categories. Feature-based methods use descriptors to match features between consecutive images [6], while direct methods seek a minimization of photometric residuals to accomplish data correlation [5,25]. Sparse optical flow tracking is an efficient direct method that is widely used [2,23,24]. It provides sub-pixel accuracy but contains more outliers than feature descriptor matching [29]. An optimization-based back-end would eliminate outliers during iteration [30]. Filter-based back-ends are meanwhile vulnerable to the outliers if only one-off updating is applied [23]. Using an iterated update scheme would mitigate this situation while introducing additive computation [31].

To recap, in order to make VIO algorithms more practical, it is desirable to develop algorithms with lower computation while maintaining high precision.

In this paper, we developed a filter-based monocular visual-inertial odometry which can be regarded as a member of MSCKF family, giving consideration to both high precision and computation efficiency. The main contributions of this paper are as follows:

- We deduced a closed-form IMU error state transition equation based on the more cognitive Hamilton notation of quaternion. By solving integration terms analytically, a novel fully linear formulation was further obtained, which is also closed-form, and furthermore, is readily implemented.

- By analyzing the statistical properties of ORB descriptor [32] distances of matched and unmatched feature points, we introduced a novel descriptor-assisted sparse optical flow tracking technique, which enhances the feature tracking robustness and barely adds any computation complexity.
- More improvements are made to improve the usability and performance of the filter. An initialization procedure is developed that automatically detects stationary scenes by analyzing tracked features and initializes the filter state based on static IMU data. The feature triangulation mechanism is carefully refined to provide efficient measurement updates.
- A filter-based monocular VIO using the proposed state transition equation, visual front-end, and initialization procedure under Sun et al.'s [24] framework is implemented. The performances of our VIO and MSCKF-MONO [23], an open-source monocular implementation of MSCKF, are compared with parameters setup as similarly as possible. Ours is also compared with other state-of-the-art open-source VIOs including ROVIO [5], OKVIS [6], and VINS-MONO [2]. In addition, we analyze the process time of our algorithm. All of the evaluations above are done on EuRoC datasets [33]. Detailed evaluations are reported.

The rest of this paper is organized as follows. The problem of quaternion notation confusion is illustrated in Section 2. Section 3 deduces the error state differential equation based on Hamilton's notation. Section 4 gives a closed-form error state transition formulation and then solves the integration terms in it, obtaining a fully linear closed-form formulation. Section 5 presents the descriptor-assisted sparse optical flow tracking front-end. Other implementation details and improvements are presented in Section 6, including the overall filter model, automatic initialization procedure, and refined feature triangulation mechanism. Section 7 presents the experimental results in detail. Finally, conclusions are made in Section 8.

2. Quaternion Notation Confusion

Quaternion is one of the widely used representations of rotation in numerical calculations [34]. In the related literature, there are mainly two different notations: Hamilton's notation and Shuster's notation [35]. The difference between them lies in their flipped rule for the multiplication of imaginary parts $\mathbf{i}$, $\mathbf{j}$, and $\mathbf{k}$. Hamilton utilizes $\mathbf{ij} = \mathbf{k}$, while Shuster advocates for $\mathbf{ij} = -\mathbf{k}$ to maintain the order of chain rule when transferring to direction cosine matrices (DCMs). Sommer et al. [28] surveyed this notation confusion problem in detail and argue for entirely abandoning Shuster's notation. In this section, we present the original problem that Shuster's notation is designed to solve and a solution for maintaining chain rule order while still using Hamilton's notation.

A quaternion of rotation $\mathbf{q}$ is basically a unit quaternion; it can be defined as

$$\mathbf{q} = \cos\frac{\theta}{2} + \mathbf{u}^A \cdot \sin\frac{\theta}{2} \tag{1}$$

where $\mathbf{u}^A$ is the unit vector of rotation axis in frame A, and θ is the angle of rotation. In the rest of this article, the term "quaternion" will be used to refer to a quaternion of rotation, for the sake of simplicity.

Equation (1) shows how to construct a quaternion $\mathbf{q}$ from an axis-angle $\theta\mathbf{u}^A$, which describes the anticlockwise rotation of an angle θ about the axis $\mathbf{u}$. If the original frame A is rotated to a new frame B after this rotation, as illustrated in Figure 1, then we can use a quaternion $\mathbf{q}_A^B$ or a DCM $\mathbf{R}_A^B$ to describe this rotation.

Note that $\mathbf{R}_A^B$ can be used to compute the coordinate of a vector $\mathbf{v}$ in frame B given its coordinate in frame A, that is $\mathbf{v}^B = \mathbf{R}_A^B\mathbf{v}^A$. $\mathbf{R}_A^B$ can be written as a function of $\mathbf{q}_A^B$

$$\mathbf{R}_A^B = C_S\left(\mathbf{q}_A^B\right) \tag{2}$$

where $C_S(\bullet)$ is an operator mapping $\mathbf{q}_A^B$ to $\mathbf{R}_A^B$.

Let $\mathbf{q}_B^C$ be the quaternion describing the rotation from frame B to frame C and $\mathbf{q}_A^C$ the rotation from frame A to frame C. Then, according to Equation (2), we have

$$\begin{aligned} \mathbf{R}_B^C &= C_S\left(\mathbf{q}_B^C\right) \\ \mathbf{R}_A^C &= C_S\left(\mathbf{q}_A^C\right) . \end{aligned} \tag{3}$$

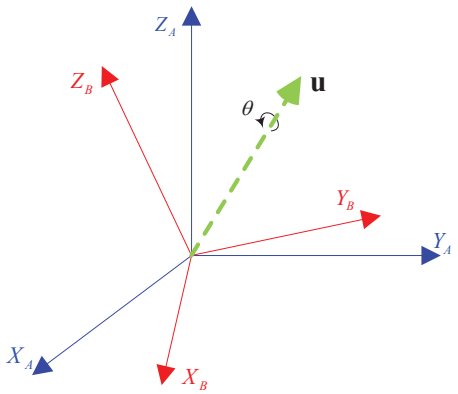

Figure 1. Rotation of frame A into frame B.

Coordinate transformation of vectors can also be done by applying the triple product of quaternions:

$$\begin{aligned} \mathbf{v}^B &= \left(\mathbf{q}_A^B\right)^{-1} \otimes \mathbf{v}^A \otimes \mathbf{q}_A^B \\ \mathbf{v}^C &= \left(\mathbf{q}_B^C\right)^{-1} \otimes \mathbf{v}^B \otimes \mathbf{q}_B^C . \end{aligned} \tag{4}$$

Here we abuse the notation of $\mathbf{v}^A$, $\mathbf{v}^B$, and $\mathbf{v}^C$ to describe quaternions with zero real part such that $\mathbf{v}^A \rightarrow \begin{bmatrix} 0 & \left(\mathbf{v}^A\right)^T \end{bmatrix}^T$. Combining the two equations above yields:

$$\begin{aligned} \mathbf{v}^C &= \left(\mathbf{q}_B^C\right)^{-1} \otimes \left(\mathbf{q}_A^B\right)^{-1} \otimes \mathbf{v}^A \otimes \mathbf{q}_A^B \otimes \mathbf{q}_B^C \\ &= \left(\mathbf{q}_A^B \otimes \mathbf{q}_B^C\right)^{-1} \otimes \mathbf{v}^A \otimes \left(\mathbf{q}_A^B \otimes \mathbf{q}_B^C\right) . \end{aligned} \tag{5}$$

Referring to Equation (5), there is

$$\mathbf{q}_A^C = \mathbf{q}_A^B \otimes \mathbf{q}_B^C . \tag{6}$$

At the same time, by applying the chain rule in DCM production, it follows that

$$\mathbf{R}_A^C = \mathbf{R}_B^C \mathbf{R}_A^B . \tag{7}$$

Now we can conclude that $C_S\left(\mathbf{q}_A^B \otimes \mathbf{q}_B^C\right) = C_S\left(\mathbf{q}_B^C\right) C_S\left(\mathbf{q}_A^B\right)$, which means the mapping $C_S(\bullet)$ is not a homomorphism. One would prefer a homomorphic mapping between DCM and quaternion to maintain the chain-rule order, which is convenient to manipulate. Shuster utilized a flipped multiplication rule to avoid this problem. This notation was adopted by the Jet Propulsion Laboratory (JPL) and thus introduced to spacecraft literatures, while other research fields were still using the traditional Hamilton notation. But as researchers have exchanged ideas between different research fields, Shuster's notation has been utilized in robotics for rotation representation [28]. So far, all of the theories about MSCKF are deduced based on this notation [1].

As Sommer et al. [28] claimed, a homomorphic mapping could be obtained even under Hamilton's notation. Let $C_H(\bullet)$ be an operator that satisfies $C_H(\mathbf{q}) = C_S(\mathbf{q})^T$. Thus, we have

$$\begin{aligned}\mathbf{R}_B^A =& C_H\left(\mathbf{q}_A^B\right)\\ \mathbf{R}_C^B =& C_H\left(\mathbf{q}_B^C\right)\\ \mathbf{R}_C^A =& C_H\left(\mathbf{q}_A^C\right).\end{aligned} \tag{8}$$

According to Equations (6) and (7), we now have $C_H\left(\mathbf{q}_A^B \otimes \mathbf{q}_B^C\right) = C_H\left(\mathbf{q}_A^B\right) C_H\left(\mathbf{q}_B^C\right)$, which proves $C_H(\bullet)$ to be a homomorphism.

Given a quaternion

$$\begin{aligned}\mathbf{q}_A^B =& \cos\frac{\theta}{2} + \mathbf{u}^A \sin\frac{\theta}{2}\\ =& q_0 + q_1\mathbf{i} + q_1\mathbf{j} + q_2\mathbf{k},\end{aligned} \tag{9}$$

the operator $C_H(\bullet)$ is defined as a function mapping quaternion $\mathbf{q}_A^B$ to a DCM $\mathbf{R}_B^A$ as

$$\begin{aligned}C_H\left(\mathbf{q}_A^B\right) =& \mathbf{R}_B^A\\ =& \begin{bmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2)\\ 2(q_1q_2+q_0q_3) & q_0^2-q_1^2+q_2^2-q_3^2 & 2(q_2q_3-q_0q_1)\\ 2(q_1q_3-q_0q_2) & 2(q_2q_3+q_0q_1) & q_0^2-q_1^2-q_2^2+q_3^2. \end{bmatrix}\end{aligned} \tag{10}$$

This is, in fact, the classical Rodrigues Rotation Formula. There is a more thorough discussion about this mapping in [36].

3. IMU Error State Differential Equation

In this section, we deduce the IMU error state differential equation based on Hamilton's quaternion notation. The Earth's rotation is ignored as low cost gyros cannot measure it. The static world assumption is employed, which means that gravity has a fixed direction. This is acceptable when a VIO is working in a limited region.

3.1. Notation

The east-north-up geographic coordinate system at initial position is selected as the reference world frame w. As the Earth's rotation is omitted, w can be regarded as an inertial frame. Quaternion $\mathbf{q}_w^b$ is used to represent the rotation from frame w to body frame b. According to Equation (8), we obtain

$$C_H\left(\mathbf{q}_w^b\right) = \mathbf{R}_b^w. \tag{11}$$

The error quaternion is defined as

$$\delta\mathbf{q}_w^b = \mathbf{q}_w^b \otimes \left(\hat{\mathbf{q}}_w^b\right)^{-1}, \tag{12}$$

where $\hat{\mathbf{q}}_w^b$ is the estimated quaternion of $\mathbf{q}_w^b$.

According to Equation (10), applying the $C_H(\bullet)$ mapping to Equation (12) leads to

$$\mathbf{R}_{w'}^w = \mathbf{R}_b^w\left(\mathbf{R}_b^{w'}\right)^{-1}, \tag{13}$$

where w' is the estimated world frame, and $\delta\mathbf{q}_w^b$ corresponds to the rotation between w and w'.

$\delta\mathbf{q}_w^b$ can be expressed in axis-angle formulation as

$$\delta \mathbf{q}_w^b = \cos \frac{|\delta \boldsymbol{\theta}^w|}{2} + \frac{\delta \boldsymbol{\theta}^w}{|\delta \boldsymbol{\theta}^w|} \sin \frac{|\delta \boldsymbol{\theta}^w|}{2}, \tag{14}$$

where $\delta \boldsymbol{\theta}^w = \begin{bmatrix} \delta\theta_x^w & \delta\theta_y^w & \delta\theta_z^w \end{bmatrix}^T$ is an axis-angle in frame w that rotates frame w' to frame w. As $|\delta \boldsymbol{\theta}^w|$ is a small angle, an approximate expression of Equation (14) is

$$\delta \mathbf{q}_w^b \approx 1 + \frac{1}{2} \delta \boldsymbol{\theta}^w. \tag{15}$$

Based on Equations (10) and (14), an approximate expression of $\mathbf{R}_{w'}^w$ is formulated as

$$\begin{aligned} \mathbf{R}_{w'}^w \approx & \begin{bmatrix} 1 & -\delta\theta_z^w & \delta\theta_y^w \\ \delta\theta_z^w & 1 & -\delta\theta_x^w \\ -\delta\theta_y^w & \delta\theta_x^w & 1 \end{bmatrix} \\ = & \mathbf{I} + [\delta \boldsymbol{\theta}^w \times], \end{aligned} \tag{16}$$

where the operator $[\bullet \times]$ is used to denote the skew matrix. For a given three-dimensional (3D) vector $\mathbf{v} = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^T$, its skew matrix is

$$[\mathbf{v} \times] = \begin{bmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ -v_y & v_x & 0 \end{bmatrix}. \tag{17}$$

3.2. IMU Measurement Model

An IMU includes a 3-axis gyroscope and a 3-axis accelerometer, whose axes are aligned with the body frame. The output of the gyroscope is modeled as

$$\boldsymbol{\omega}_m^b = \boldsymbol{\omega}_{wb}^b + \mathbf{b}_g + \mathbf{n}_g, \tag{18}$$

where $\boldsymbol{\omega}_{wb}^b$ is the true angular velocity, $\mathbf{b}_g$ denotes the gyroscope bias under the body frame, and $\mathbf{n}_g$ is the Gaussian white noise.

The accelerometer measures the specific force along a body-fixed axis, which includes an opposite gravity and is affected by bias and noise as well:

$$\begin{aligned} \mathbf{f}_m^b = & \ \mathbf{a}^b - \mathbf{R}_w^b \mathbf{g}^w + \mathbf{b}_a + \mathbf{n}_a \\ = & \ \mathbf{R}_w^b \left(\mathbf{a}^w - \mathbf{g}^w \right) + \mathbf{b}_a + \mathbf{n}_a, \end{aligned} \tag{19}$$

where $\mathbf{a}^b$ is the true acceleration, and $\mathbf{g}^w = \begin{bmatrix} 0 & 0 & -g \end{bmatrix}^T$ denotes the gravity under the world frame. $\mathbf{b}_a$ and $\mathbf{n}_a$ denote the bias and the Gaussian white noise under the body frame, respectively.

The biases $\mathbf{b}_g$ and $\mathbf{b}_a$ are modeled as random walk processes

$$\begin{aligned} \dot{\mathbf{b}}_g = & \mathbf{n}_{wg} \\ \dot{\mathbf{b}}_a = & \mathbf{n}_{wa}, \end{aligned} \tag{20}$$

where $\mathbf{n}_{wg}$ and $\mathbf{n}_{wa}$ are Gaussian white noises.

3.3. IMU Error State Definition

The IMU state includes the quaternion $\mathbf{q}_w^b$, velocity $\mathbf{v}_b^w$ and position $\mathbf{p}_b^w$ of the body frame origin in the world frame, and IMU biases $\mathbf{b}_g$ and $\mathbf{b}_a$. The IMU state can be defined as

$$\mathbf{x}_{IMU} = \begin{bmatrix} \mathbf{q}_w^{b\,T} & \mathbf{v}_b^{wT} & \mathbf{p}_b^{wT} & \mathbf{b}_g^T & \mathbf{b}_a^T \end{bmatrix}^T. \tag{21}$$

The filter is designed based on the error state because it is convenient to process by extended Kalman filter (EKF). Three dimensional angular error $\delta\boldsymbol{\theta}^w$ rather than four dimensional quaternion error $\delta\mathbf{q}_w^b$ is utilized since it is accordance with the degree of freedom (DOF) of rotation, and thus a minimum parameterization.

Other error state components are simply defined as the Euclidean distances between true states and the estimated states, which lead to

$$\delta\mathbf{v}_b^w = \mathbf{v}_b^w - \hat{\mathbf{v}}_b^w, \tag{22}$$
$$\delta\mathbf{p}_b^w = \mathbf{p}_b^w - \hat{\mathbf{p}}_b^w, \tag{23}$$
$$\delta\mathbf{b}_g = \mathbf{b}_g - \hat{\mathbf{b}}_g, \tag{24}$$
$$\delta\mathbf{b}_a = \mathbf{b}_a - \hat{\mathbf{b}}_a. \tag{25}$$

The overall IMU error state can now be concluded as

$$\delta\mathbf{x}_{IMU} = \begin{bmatrix} \delta\boldsymbol{\theta}^{wT} & \delta\mathbf{v}_b^{wT} & \delta\mathbf{p}_b^{wT} & \mathbf{b}_g^T & \mathbf{b}_a^T \end{bmatrix}^T. \tag{26}$$

3.4. Differential Equation

The matrix form of the differential equation of the overall IMU error state is as follows.

$$\delta\dot{\mathbf{x}}_{IMU} = \mathbf{F}\delta\mathbf{x}_{IMU} + \mathbf{G}\mathbf{n}_{IMU}, \tag{27}$$

where $\mathbf{n}_{IMU}$ denotes the IMU noise, given by

$$\delta\mathbf{x}_{IMU} = \begin{bmatrix} \delta\theta^w & \delta\mathbf{v}_b^{wT} & \delta\mathbf{p}_b^{wT} & \delta\mathbf{b}_g^T & \delta\mathbf{b}_a^T \end{bmatrix}^T, \tag{28}$$
$$\mathbf{n}_{IMU} = \begin{bmatrix} \mathbf{n}_g^T & \mathbf{n}_a^T & \mathbf{n}_{wg}^T & \mathbf{n}_{wa}^T \end{bmatrix}^T, \tag{29}$$

and the matrices $\mathbf{F}$ and $\mathbf{G}$ are as follows:

$$\mathbf{F} = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -\hat{\mathbf{R}}_b^w & \mathbf{0}_{3\times3} \\ -\left[\left(\hat{\mathbf{R}}_b^w\hat{\mathbf{a}}\right)\times\right] & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -\hat{\mathbf{R}}_b^w \\ \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{bmatrix}, \tag{30}$$

$$\mathbf{G} = \begin{bmatrix} -\hat{\mathbf{R}}_b^w & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & -\hat{\mathbf{R}}_b^w & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} \end{bmatrix}. \tag{31}$$

4. Fully Linear State Transition Equation Formulation

A state transition equation is needed for the extended Kalman filter (EKF) to propagate the state and covariance. One commonly used method is to make a first-order approximation based on a continuous differential equation [37]. Li and Mourikis [19] proposed a closed-form error state transition equation that effectuated a system model with no information loss. However, there are still some tricky integration terms left behind. In this section, we first present the closed-form IMU error

state transition equation based on the results of Section 3. Then, we solve the integration terms by two-sample fitting of the rotation matrix, resulting in a closed-form formulation that is fully linear.

4.1. Original Closed-Form Equation

Following the methodology of Li and Mourikis [19], the closed-form transition equation was deduced and presented in what follows. Noting that k and $k+1$ are consecutive discrete sampling instants of IMU, and Δt is the sampling period,

$$\delta\boldsymbol{\theta}^{w}_{k+1|k} = \boldsymbol{\Phi}_{\theta\theta}\left(k+1,k\right)\delta\boldsymbol{\theta}^{w}_{k|k} + \boldsymbol{\Phi}_{\theta b_g}\left(k+1,k\right)\delta\mathbf{b}_{gk|k} + \mathbf{n}_{\theta k}, \tag{32}$$

$$\begin{aligned}\delta\mathbf{v}^{w}_{k+1|k} =& \boldsymbol{\Phi}_{v\theta}\left(k+1,k\right)\delta\boldsymbol{\theta}^{w}_{k|k} + \boldsymbol{\Phi}_{vv}\left(k+1,k\right)\delta\mathbf{v}^{w}_{k|k} \\ &+ \boldsymbol{\Phi}_{vb_g}\left(k+1,k\right)\delta\mathbf{b}_{gk|k} + \boldsymbol{\Phi}_{vb_a}\left(k+1,k\right)\delta\mathbf{b}_{ak|k} + \mathbf{n}_{vk},\end{aligned} \tag{33}$$

$$\begin{aligned}\delta\mathbf{p}^{w}_{k+1|k} =& \boldsymbol{\Phi}_{p\theta}\left(k+1,k\right)\delta\boldsymbol{\theta}^{w}_{k|k} + \boldsymbol{\Phi}_{pv}\left(k+1,k\right)\delta\mathbf{v}^{w}_{k|k} + \boldsymbol{\Phi}_{pp}\left(k+1,k\right)\delta\mathbf{p}^{w}_{k|k} \\ &+ \boldsymbol{\Phi}_{pb_g}\left(k+1,k\right)\delta\mathbf{b}_{gk|k} + \boldsymbol{\Phi}_{pb_a}\left(k+1,k\right)\delta\mathbf{b}_{ak|k} + \mathbf{n}_{pk},\end{aligned} \tag{34}$$

$$\delta\mathbf{b}_{gk+1|k} = \boldsymbol{\Phi}_{b_gb_g}\left(k+1,k\right)\delta\mathbf{b}_{gk|k} + \mathbf{n}_{b_gk}, \tag{35}$$

$$\delta\mathbf{b}_{ak+1|k} = \boldsymbol{\Phi}_{b_ab_a}\left(k+1,k\right)\delta\mathbf{b}_{ak|k} + \mathbf{n}_{b_ak}, \tag{36}$$

where $\boldsymbol{\Phi}_{x_1x_2}$ is used to represent the transition matrix of the error state of x_1 with respect to the error state of x_2, and $\mathbf{n}_*$ terms represent noise. All of the $\boldsymbol{\Phi}_*$ and $\mathbf{n}_*$ terms are listed as follows:

$$\boldsymbol{\Phi}_{\theta\theta}\left(k+1,k\right) = \mathbf{I}_{3\times3}, \tag{37}$$

$$\boldsymbol{\Phi}_{\theta b_g}\left(k+1,k\right) = -\hat{\mathbf{R}}^{w}_{b_k}\int_{t_k}^{t_{k+1}}\hat{\mathbf{R}}^{b_k}_{b_\tau}d\tau, \tag{38}$$

$$\boldsymbol{\Phi}_{v\theta}\left(k+1,k\right) = -\left[\left(\hat{\mathbf{v}}^{w}_{k+1|k} - \hat{\mathbf{v}}^{w}_{k|k} - \mathbf{g}^{w}\Delta t\right)\times\right], \tag{39}$$

$$\boldsymbol{\Phi}_{vv}\left(k+1,k\right) = \mathbf{I}_{3\times3}, \tag{40}$$

$$\boldsymbol{\Phi}_{vb_g}\left(k+1,k\right) = \int_{t_k}^{t_{k+1}}\left\{\left[\left(\hat{\mathbf{v}}^{w}_{\tau} - \mathbf{g}^{w}\right)\times\right]\hat{\mathbf{R}}^{w}_{b_k}\left(\int_{t_k}^{\tau}\hat{\mathbf{R}}^{b_k}_{b_m}dm\right)\right\}d\tau, \tag{41}$$

$$\boldsymbol{\Phi}_{vb_a}\left(k+1,k\right) = -\hat{\mathbf{R}}^{w}_{b_k}\int_{t_k}^{t_{k+1}}\hat{\mathbf{R}}^{b_k}_{b_\tau}d\tau, \tag{42}$$

$$\boldsymbol{\Phi}_{p\theta}\left(k+1,k\right) = -\left[\left(\hat{\mathbf{p}}^{w}_{k+1|k} - \hat{\mathbf{p}}^{w}_{k|k} - \hat{\mathbf{v}}^{w}_{k|k}\Delta t - \frac{1}{2}\mathbf{g}^{w}\Delta t^2\right)\times\right], \tag{43}$$

$$\boldsymbol{\Phi}_{pv}\left(k+1,k\right) = \mathbf{I}_{3\times3}\Delta t, \tag{44}$$

$$\boldsymbol{\Phi}_{pp}\left(k+1,k\right) = \mathbf{I}_{3\times3}, \tag{45}$$

$$\boldsymbol{\Phi}_{pb_g}\left(k+1,k\right) = \int_{t_k}^{t_{k+1}}\left\{\int_{t_k}^{\tau}\left(\left[\left(\hat{\mathbf{v}}^{w}_{s} - \mathbf{g}^{w}\right)\times\right]\hat{\mathbf{R}}^{w}_{b_k}\left[\int_{t_k}^{s}\hat{\mathbf{R}}^{b_k}_{b_m}dm\right]\right)ds\right\}d\tau, \tag{46}$$

$$\boldsymbol{\Phi}_{pb_a}\left(k+1,k\right) = -\hat{\mathbf{R}}^{w}_{b_k}\int_{t_k}^{t_{k+1}}\left(\int_{t_k}^{\tau}\hat{\mathbf{R}}^{b_k}_{b_s}ds\right)d\tau, \tag{47}$$

$$\boldsymbol{\Phi}_{b_gb_g}\left(k+1,k\right) = \mathbf{I}_{3\times3}, \tag{48}$$

$$\boldsymbol{\Phi}_{b_ab_a}\left(k+1,k\right) = \mathbf{I}_{3\times3}, \tag{49}$$

$$\mathbf{n}_{\theta k} = \hat{\mathbf{R}}^{w}_{b_k}\int_{t_k}^{t_{k+1}}\hat{\mathbf{R}}^{b_k}_{b_\tau}\left(\int_{t_k}^{\tau}\mathbf{n}_{wgs}ds + \mathbf{n}_{g\tau}\right)d\tau, \tag{50}$$

$$\begin{aligned}\mathbf{n}_{vk} =& \int_{t_k}^{t_{k+1}}\hat{\mathbf{R}}^{w}_{b_\tau}\left(-\int_{t_k}^{\tau}\mathbf{n}_{was}ds - \mathbf{n}_{a\tau}\right)d\tau \\ &- \int_{t_k}^{t_{k+1}}\left\{\left[\left(\hat{\mathbf{v}}^{w}_{\tau} - \mathbf{g}^{w}\right)\times\right]\hat{\mathbf{R}}^{w}_{b_k}\left[\int_{t_k}^{\tau}\hat{\mathbf{R}}^{b_k}_{b_m}\left(\int_{t_k}^{m}\mathbf{n}_{wgs}ds + \mathbf{n}_{gm}\right)dm\right]\right\}d\tau,\end{aligned} \tag{51}$$

$$\mathbf{n}_{pk} = \int_{t_k}^{t_{k+1}} \mathbf{n}_{v\tau} d\tau, \tag{52}$$

$$\mathbf{n}_{b_g k} = \int_{t_k}^{t_{k+1}} \mathbf{n}_{wg\tau} d\tau, \tag{53}$$

$$\mathbf{n}_{b_a k} = \int_{t_k}^{t_{k+1}} \mathbf{n}_{wa\tau} d\tau. \tag{54}$$

4.2. Fully Linear Closed-Form Formulation

Notice that in Equations (38), (41), (42), (46), and (47), although they are closed-form expressions, there are still some tricky integration terms that are not straightforward for implementation. One can solve these terms with numerical integration, but here we present a fully linear analytical expression that is readily implemented. The key is to solve $\int_{t_k}^{t_{k+1}} \hat{\mathbf{R}}_{b_\tau}^{b_k} d\tau$. We first apply a two-sample fitting method to approximate the axis-angle representing $\hat{\mathbf{R}}_{b_\tau}^{b_k}$, then Rodrigues' rotation formula is applied to express the DCM as a linear function of τ, thus making the integration easy to solve.

4.2.1. Two-Sample Fitting of Axis-Angle

Any DCM can be regarded as a single rotation about a fixed axis, and thus can be represented by an axis-angle. Let the axis-angle of $\hat{\mathbf{R}}_{b_{k+1}}^{b_k}$ be $\boldsymbol{\phi} = \alpha \mathbf{u}^{b_k}$, where α is the angle of rotation and $\mathbf{u}^{b_k}$ is the rotation axis. Let τ be a time instant between t_k and t_{k+1} and $\varepsilon = \tau - t_k$, then a linear model can be used to represent the axis-angle of $\hat{\mathbf{R}}_{b_\tau}^{b_k}$:

$$\boldsymbol{\phi}_\tau = \boldsymbol{\phi}(\varepsilon) = \frac{\varepsilon}{\Delta t} \alpha \mathbf{u}^{b_k} = \frac{\varepsilon}{\Delta t} \boldsymbol{\phi}. \tag{55}$$

Angular velocity measurements of t_k and t_{k+1} are available when calculating the transition matrix $\boldsymbol{\Phi}(k+1, k)$, so a two-sample fitting method can be used to approximate the axis-angle $\boldsymbol{\phi}$. We start from the differential equation of $\boldsymbol{\phi}(\varepsilon)$ [36]:

$$\dot{\boldsymbol{\phi}}(\varepsilon) = \bar{\boldsymbol{\omega}} + \frac{1}{2} \boldsymbol{\phi}(\varepsilon) \times \bar{\boldsymbol{\omega}} + \frac{1}{12} \boldsymbol{\phi}(\varepsilon) \times (\boldsymbol{\phi}(\varepsilon) \times \bar{\boldsymbol{\omega}}), \tag{56}$$

where $\bar{\boldsymbol{\omega}}$ is the average angular velocity between t_k and t_{k+1}. As two gyro measurements are available, we use a straight line model to fit $\bar{\boldsymbol{\omega}}$ as

$$\bar{\boldsymbol{\omega}}(t_k + \varepsilon) = \mathbf{a} + 2\mathbf{b}\varepsilon, \quad 0 \le \varepsilon \le \Delta t. \tag{57}$$

Considering $\bar{\boldsymbol{\omega}}^b(t_k) = \hat{\boldsymbol{\omega}}_{wb}^b(t_k)$ and $\bar{\boldsymbol{\omega}}^b(t_k + \Delta t) = \hat{\boldsymbol{\omega}}_{wb}^b(t_{k+1})$, leads to

$$\begin{aligned} \mathbf{a} =& \hat{\boldsymbol{\omega}}_{wb}^b(t_k) \\ \mathbf{b} =& \left(\hat{\boldsymbol{\omega}}_{wb}^b(t_{k+1}) - \hat{\boldsymbol{\omega}}_{wb}^b(t_k) \right) \Big/ (2\Delta t). \end{aligned} \tag{58}$$

According to Equation (55), $\boldsymbol{\phi}$ is equal to $\boldsymbol{\phi}_{t_{k+1}}$, then using Taylor expansion to expand $\boldsymbol{\phi}$ at linearized point t_k yields

$$\begin{aligned} \boldsymbol{\phi} =& \boldsymbol{\phi}_{t_{k+1}} \\ =& \boldsymbol{\phi}_{t_k} + \Delta t \dot{\boldsymbol{\phi}}_{t_k} + \frac{\Delta t^2}{2!} \ddot{\boldsymbol{\phi}}_{t_k} + \cdots \\ =& \boldsymbol{\phi}(0) + \Delta t \dot{\boldsymbol{\phi}}(0) + \frac{\Delta t^2}{2!} \ddot{\boldsymbol{\phi}}(0) + \cdots. \end{aligned} \tag{59}$$

Now define a new function of ε as

$$\Delta\boldsymbol{\theta}\left(\varepsilon\right)=\int_{0}^{\varepsilon}\hat{\boldsymbol{\omega}}_{wb}^{b}\left(t_{k}+\varepsilon\right)d\varepsilon. \tag{60}$$

It can be pointed out that $\boldsymbol{\phi}\left(\varepsilon\right)\approx\Delta\boldsymbol{\theta}\left(\varepsilon\right)$. Derivatives of $\Delta\boldsymbol{\theta}\left(0\right)$ are defined as

$$\begin{aligned}\Delta\boldsymbol{\theta}\left(0\right)=&\mathbf{0},\\ \Delta\dot{\boldsymbol{\theta}}\left(0\right)=&\hat{\boldsymbol{\omega}}_{wb}^{b}\left(t_{k}\right)=\mathbf{a},\\ \Delta\ddot{\boldsymbol{\theta}}\left(0\right)=&\dot{\hat{\boldsymbol{\omega}}}_{wb}^{b}\left(t_{k}\right)=2\mathbf{b},\\ \Delta\boldsymbol{\theta}^{(i)}\left(0\right)=&\hat{\boldsymbol{\omega}}_{wb}^{b}{}^{(i-1)}\left(t_{k}\right)=\mathbf{0},\quad i=3,4,5,\cdots.\end{aligned} \tag{61}$$

The third term in Equation (56) is a high-order small quantity that can be omitted. By substituting $\boldsymbol{\phi}\left(\varepsilon\right)$ as $\Delta\boldsymbol{\theta}\left(\varepsilon\right)$, Equation (56) turns into

$$\dot{\boldsymbol{\phi}}\left(\varepsilon\right)=\bar{\boldsymbol{\omega}}\left(t_{k}+\varepsilon\right)+\frac{1}{2}\Delta\boldsymbol{\theta}\left(\varepsilon\right)\times\bar{\boldsymbol{\omega}}\left(t_{k}+\varepsilon\right). \tag{62}$$

Now the high-order derivatives of $\boldsymbol{\phi}\left(\varepsilon\right)$ can be obtained:

$$\begin{aligned}\ddot{\boldsymbol{\phi}}\left(\varepsilon\right)=&\dot{\bar{\boldsymbol{\omega}}}\left(t_{k}+\varepsilon\right)+\frac{1}{2}\Delta\dot{\boldsymbol{\theta}}\left(\varepsilon\right)\times\bar{\boldsymbol{\omega}}\left(t_{k}+\varepsilon\right)+\frac{1}{2}\Delta\boldsymbol{\theta}\left(\varepsilon\right)\times\dot{\bar{\boldsymbol{\omega}}}\left(t_{k}+\varepsilon\right),\\ \boldsymbol{\phi}^{(3)}\left(\varepsilon\right)=&\frac{1}{2}\Delta\ddot{\boldsymbol{\theta}}\left(\varepsilon\right)\times\bar{\boldsymbol{\omega}}\left(t_{k}+\varepsilon\right)+\Delta\dot{\boldsymbol{\theta}}\left(\varepsilon\right)\times\dot{\bar{\boldsymbol{\omega}}}\left(t_{k}+\varepsilon\right),\\ \boldsymbol{\phi}^{(4)}\left(\varepsilon\right)=&\frac{3}{2}\Delta\ddot{\boldsymbol{\theta}}\left(\varepsilon\right)\times\dot{\bar{\boldsymbol{\omega}}}\left(t_{k}+\varepsilon\right),\\ \boldsymbol{\phi}^{(i)}\left(\varepsilon\right)=&\mathbf{0},\quad i=5,6,7,\cdots.\end{aligned} \tag{63}$$

Let $\varepsilon=0$, and considering Equation (61), we have

$$\begin{aligned}\boldsymbol{\phi}\left(0\right)=&\mathbf{0},\\ \dot{\boldsymbol{\phi}}\left(0\right)=&\mathbf{a},\\ \ddot{\boldsymbol{\phi}}\left(0\right)=&2\mathbf{b},\\ \boldsymbol{\phi}^{(3)}\left(0\right)=&\mathbf{a}\times\mathbf{b},\\ \boldsymbol{\phi}^{(i)}\left(0\right)=&\mathbf{0},i=4,5,6,\cdots.\end{aligned} \tag{64}$$

Substituting the equations above into Equation (59) yields

$$\begin{aligned}\boldsymbol{\phi}=&\mathbf{a}\Delta t+\mathbf{b}\Delta t^{2}+\frac{1}{6}\left(\mathbf{a}\times\mathbf{b}\right)\Delta t^{3}\\ =&\frac{1}{2}\left(\hat{\boldsymbol{\omega}}_{wb}^{b}\left(k\right)+\hat{\boldsymbol{\omega}}_{wb}^{b}\left(k+1\right)\right)\Delta t+\frac{1}{12}\left(\hat{\boldsymbol{\omega}}_{wb}^{b}\left(k\right)\times\hat{\boldsymbol{\omega}}_{wb}^{b}\left(k+1\right)\right)\Delta t^{2}.\end{aligned} \tag{65}$$

This is how the axis-angle between two consecutive sampling time instants t_k and t_{k+1} can be computed.

According to Rodrigues' rotation formula,

$$\hat{\mathbf{R}}_{b_{k+1}}^{b_{k}}=\mathbf{I}+\sin\alpha\left[\mathbf{u}^{b_{k}}\times\right]+\left(1-\cos\alpha\right)\left[\mathbf{u}^{b_{k}}\times\right]^{2}. \tag{66}$$

As α is a small angular, since Δt is small, Equation (66) has an approximation

$$\begin{aligned}\hat{\mathbf{R}}_{b_{k+1}}^{b_k} \approx & \mathbf{I} + \alpha \left[\mathbf{u}^{b_k} \times\right] \\ = & \mathbf{I} + [\boldsymbol{\phi} \times] .\end{aligned} \tag{67}$$

Now, substituting Equation (55) into Equation (67) leads to

$$\hat{\mathbf{R}}_{b_\tau}^{b_k} \approx \mathbf{I} + \frac{\tau - t_k}{\Delta t} [\boldsymbol{\phi} \times] . \tag{68}$$

Finally, the general procedure to solve the integration term $\int_{t_k}^{t_{k+1}} \hat{\mathbf{R}}_{b_\tau}^{b_k} d\tau$ can be summarized as follows:

1. Compute the axis-angle between t_k and t_{k+1} according to Equation (65).
2. Express $\hat{\mathbf{R}}_{b_\tau}^{b_k}$ as Equation (68).
3. Easily solve the $\int_{t_k}^{t_{k+1}} \hat{\mathbf{R}}_{b_\tau}^{b_k} d\tau$ term, as it becomes an integration about a linear analytic expression.

4.2.2. Solve Integration Terms in $\boldsymbol{\Phi}_*$

The fully linear closed-form transition matrix of Equations (38), (41), (42), (46), and (47) can now be obtained by simply solving the integration terms. The results are listed below.

$$\begin{aligned}
\boldsymbol{\Phi}_{\theta b_g}(k+1,k) = & -\hat{\mathbf{R}}_{b_k}^{w} \left(\Delta t \mathbf{I} + \frac{1}{2}\Delta t [\boldsymbol{\phi} \times]\right), \\
\boldsymbol{\Phi}_{v b_g}(k+1,k) = & \left[\left(-\hat{\mathbf{p}}_{k+1|k}^{w} + \hat{\mathbf{p}}_{k|k}^{w} + \hat{\mathbf{v}}_{k+1|k}^{w}\Delta t - \frac{1}{2}\mathbf{g}^{w}\Delta t^2\right) \times\right] \hat{\mathbf{R}}_{b_k}^{w} \\
& + \left[\left(-\frac{1}{2}\hat{\mathbf{p}}_{k+1|k}^{w} + \frac{1}{2}\hat{\mathbf{p}}_{k|k}^{w} + \frac{1}{2}\hat{\mathbf{v}}_{k+1|k}^{w}\Delta t - \frac{1}{6}\mathbf{g}^{w}\Delta t^2\right) \times\right] \hat{\mathbf{R}}_{b_k}^{w} [\boldsymbol{\phi} \times], \\
\boldsymbol{\Phi}_{v b_a}(k+1,k) = & -\hat{\mathbf{R}}_{b_k}^{w} \left(\Delta t \mathbf{I} + \frac{1}{2}\Delta t [\boldsymbol{\phi} \times]\right), \\
\boldsymbol{\Phi}_{p b_g}(k+1,k) = & \left[\left(-\frac{1}{6}\mathbf{g}^{w}\Delta t^3\right) \times\right] \hat{\mathbf{R}}_{b_k}^{w} \\
& + \left[\left(\frac{1}{4}\hat{\mathbf{p}}_{k+1|k}^{w}\Delta t - \frac{1}{4}\hat{\mathbf{p}}_{k|k}^{w}\Delta t - \frac{1}{24}\mathbf{g}^{w}\Delta t^3\right) \times\right] \hat{\mathbf{R}}_{b_k}^{w} [\boldsymbol{\phi} \times], \\
\boldsymbol{\Phi}_{p b_a}(k+1,k) = & -\frac{1}{6}\hat{\mathbf{R}}_{b_k}^{w}\Delta t^2 (3\mathbf{I} + [\boldsymbol{\phi} \times]) .
\end{aligned} \tag{69}$$

Notice that all of the variables needed are available at the time of calculating the $\boldsymbol{\Phi}$ terms above. This model is unbiased up to the information loss of the two-sample fitting of DCM, which is small due to the utilization of all related measurement data.

4.2.3. Process Noise Terms

The property of noise terms in Equations (50)–(54) should be acquired to compute the process noise covariance matrix in a Kalman filter. The process noise covariance at t_k can be computed as [37]:

$$\mathbf{Q}(t_k) = \int_{t_k}^{t_{k+1}} \boldsymbol{\Phi}(t_{k+1}, \tau) \mathbf{G}(\tau) \mathbf{q} \mathbf{G}^T(\tau) \boldsymbol{\Phi}^T(t_{k+1}, \tau) d\tau. \tag{70}$$

We temporarily abuse symbol $\mathbf{q}$ here to represent the noise intensity matrix. $\boldsymbol{\Phi}$ is the overal; IMU error state transition matrix. As Δt is a small quantity, an approximate expression of Equation (70) is formulated as

$$\mathbf{Q}(t_k) \approx \boldsymbol{\Phi}(t_{k+1}, t_k) \mathbf{G}(t_k) \mathbf{q} \mathbf{G}^T(t_k) \boldsymbol{\Phi}^T(t_{k+1}, t_k) \Delta t. \tag{71}$$

The discrete form, which will be preferable for a discrete filter implementation, is

$$\mathbf{Q}(k) \approx \mathbf{\Phi}(k+1,k)\,\mathbf{G}(k)\,\mathbf{q}\mathbf{G}^{T}(k)\,\mathbf{\Phi}^{T}(k+1,k)\,\Delta t. \tag{72}$$

4.3. Summarization

According to the derivation above, the proposed fully linear closed-form IMU error state transition equation is as follows:

$$\delta\mathbf{x}_{IMU}(k+1) = \mathbf{\Phi}(k+1,k)\,\delta\mathbf{x}_{IMU}(k) + \mathbf{n}_{IMU}(k), \tag{73}$$

where

$$\mathbf{\Phi}(k+1,k) = \begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{\Phi}_{\theta b_g} & \mathbf{0}_{3\times3} \\ \mathbf{\Phi}_{v\theta} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{\Phi}_{vb_g} & \mathbf{\Phi}_{vb_a} \\ \mathbf{\Phi}_{p\theta} & \Delta t\mathbf{I}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{\Phi}_{pb_g} & \mathbf{\Phi}_{pb_a} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} \end{bmatrix}, \tag{74}$$

$$\mathbf{n}_{IMU}(k) = \begin{bmatrix} \mathbf{n}_{\theta k}^{T} & \mathbf{n}_{vk}^{T} & \mathbf{n}_{pk}^{T} & \mathbf{n}_{b_g k}^{T} & \mathbf{n}_{b_a k}^{T} \end{bmatrix}^{T}, \tag{75}$$

and the covariance matrix of $\mathbf{n}_{IMU}(k)$ is $E\left[\mathbf{n}_{IMU}(k)\,\mathbf{n}_{IMU}^{T}(k)\right] = \mathbf{Q}(k)$.

The integration terms are solved using a fitting rule of DCMs by utilizing all of the related measurements, so we claim that the obtained formulation is an unbiased model up to the numerical integration resolution.

5. ORB Descriptor-Assisted Optical Flow Front-End

In this section, we propose a sparse visual front-end using descriptor-assisted optical flow feature tracking.

Different kinds of feature descriptors are used in several VIOs to accomplish feature extraction and matching [1,6,30]. In contrast, other solutions choose optical flow feature tracking as their front-end solution since it is not that time-consuming compared to the descriptor-based methods [2,23,24]. However, there are more wrong matches in optical flow tracking than in descriptor-based methods, and these wrong matches exist even after eliminating algorithms such as random sample consensus (RANSAC). Filter-based VIOs are very sensitive to feature outliers since they don't eliminate outliers in their iterations as the optimization-based ones do. Wrong matches left behind will participate in measurement updates, which may result in deteriorating estimates or even failure. As a conclusion, a robust front-end is needed to achieve stable performance for filter-based VIOs, while a real-time solution also calls for fast data correlation.

Yang et al. [29], refined ORB-SLAM [38] by using a sparse optical flow algorithm. The key idea was to correct the image coordinates of ORB features by optical flow tracking results to achieve sub-pixel precision. The proposed method here is a bit different since we use optical flow to first conduct a fast tracking, then compute descriptor distance between matched feature pair members and justify whether they are a good match-up.

There exist plenty of feature descriptor algorithms. We chose the ORB descriptor in our proposed method for two reasons:

1. The ORB descriptor is a binary string, so the distance between two descriptors can be expressed as a Hamming distance, which can be computed efficiently.
2. The rotation between consecutive images in a real-time application is usually very gentle, so invariance to rotation is not very important for a descriptor.

Descriptor Distance Analysis for General Corner Features

The basic visual front-end is based on Shi-Tomasi corner detection [39] and optical flow tracking [40]. It is important to figure out whether the ORB descriptor is meaningful for general Shi-Tomasi corner features. An experiment was done and proved that it is indeed meaningful statistically. We calculated the feature angle for a Shi-Tomasi feature and then used it to compute the ORB descriptor [32]. Several tests were conducted in the experiment. For each test, feature pairs from every two adjacent images of a continuous image stream were stored separately in two sequences. These tests basically analyzed the statistical properties of ORB descriptor distances of feature pairs, including

1. Coarsely matched feature pairs based on Shi-Tomasi corner detection and optical flow tracking.
2. Relatively strictly matched feature pairs based on ORB descriptor matching and RANSAC.
3. Randomly constructed feature pairs.
4. Unmatched feature pairs generated by inverse order of one of the strictly matched feature sequences.

One feature sequence from strictly matched pairs was inverted to generate strictly unmatched feature pairs. The experimental result is shown in Figure 2.

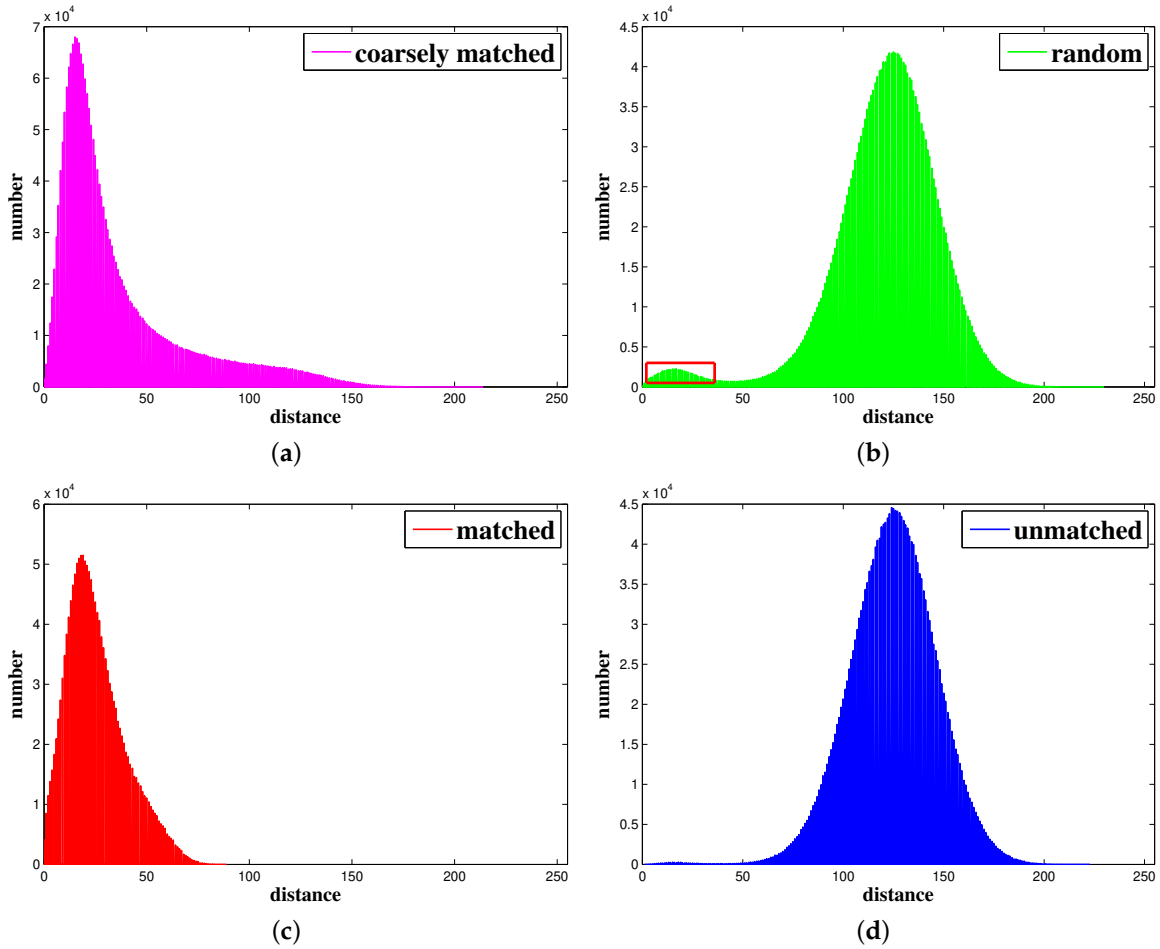

Figure 2. Statistical distribution of ORB descriptor [32] distances for coarsely matched, strictly matched, random constructed, and unmatched Shi-Tomasi feature pairs. The X axis represents descriptor distances and ranges from 0 to 255. The range of the Y axis is determined by the number of feature pairs in each experiment. (**a**) Coarsely matched features results. (**b**) Random constructed features results. (**c**) Strictly matched features results. (**d**) Unmatched features results.

We strongly suspect the very long tail in Figure 2a may be due to wrong matches because no further outlier rejection method was applied after optical flow tracking in this test. In Figure 2b, except for the massive Guassian-like distribution, a little bump centered at about 17 appeared, which is framed by a red rectangular border. This is because the random pairs were constructed in two adjacent images and thus, two matched features have a considerable probability of being coincidentally formed into a pair. These two experiments prove that ORB descriptors and descriptor distances are meaningful for general Shi-Tomasi corners, from a statistical standpoint.

In order to clearly analyze the statistical properties of matched and unmatched pairs, two further tests were conducted. First, a descriptor-based matching and RANSAC mechanism were applied to obtain relatively strictly matched feature pairs. Then, the order of one of the feature sequences was reversed, which is a simple yet effective way to make two sequences unmatched. Descriptor distances before and after order reversion were computed, and statistical results are shown in Figure 2c,d. It can be seen from the figures that the long tail and little bump disappear because of the relatively strict pairing rule. They are plotted together in Figure 3 to make a clear comparison.

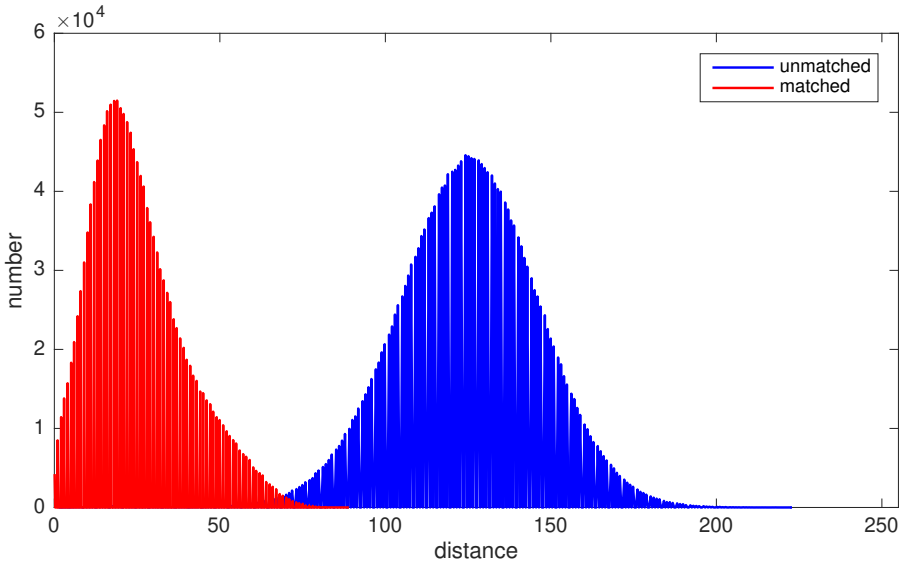

Figure 3. This figure shows the descriptor distances of unmatched and matched feature pairs. It can be clearly seen that the difference is statistically significant, thus a heuristic algorithm can be used to pick out outliers.

The experimental results show that the descriptor distances of unmatched and matched feature pairs possess significantly different statistical properties. As shown in Figure 3, descriptor distances of unmatched features approximately follow the Gaussian distribution with a mean, or we can say peak, at about 124.7 and with a standard deviation of 21.8. For matched pairs, the distribution shows a sharper peak at about 18.5. There is still a tail in the matched distribution, but it is much smaller than the one in Figure 2a. The difference between matched and unmatched pairs is significant enough to design a strategy to filter out wrong matches.

We use a heuristic to complete the mission:

- For feature pairs with distances lower than the smaller peak value, classify them as inliers.
- For feature pairs with distances higher than the bigger peak value, classify them as outliers.
- For feature pairs whose distances are between two peaks, calculate and compare the Mahalanobis distances to both peak to decide their classification.

6. EKF-Based VIO Implementation Details and Improvements

In this section, implementation details and improvements of the proposed EKF-based VIO are presented, including filtering scheme, automatic initialization procedure, and refined feature

triangulation mechanism. An overall flow chart of the implemented VIO algorithm is shown in Figure 4. Red sections highlight novelties proposed in this paper.

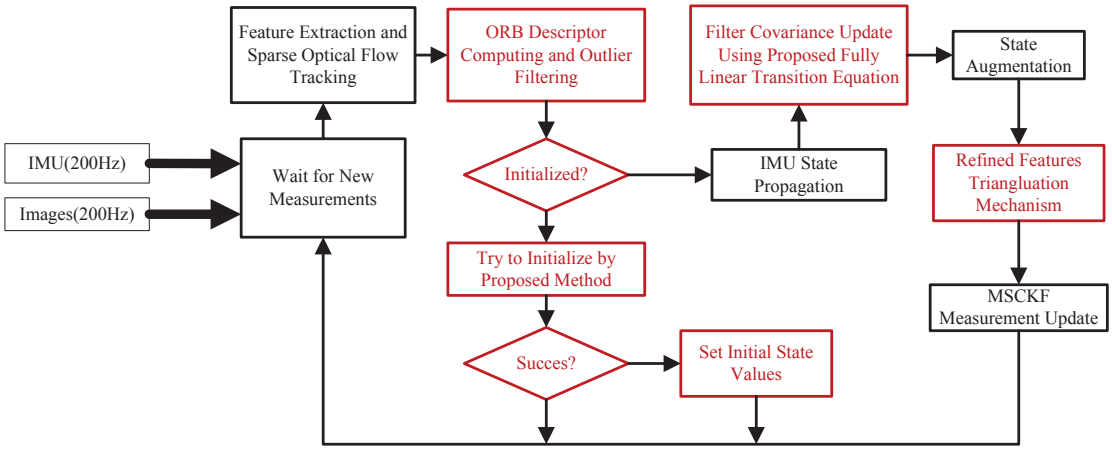

Figure 4. Flow chart of extended Kalman filter (EKF)-based visual-inertial odometry (VIO) implementation. Red sections highlight novelties proposed in this paper. Term "IMU" stands for inertial measurement unit, and term "MSCKF" stands for multi-state constraint Kalman filter.

6.1. Filter State and Measurement Model

A VIO following the scheme of Mourikis and Roumeliotis [1] is implemented. The system state includes a sliding window of N historical IMU poses and camera-IMU extrinsic as proposed by Li and Mourikis [22]. The overall system state is formulated as

$$\mathbf{x}_{all} = \begin{bmatrix} \mathbf{x}_{IMU} & \mathbf{q}_b^c & \mathbf{p}_c^b & \mathbf{q}_w^{b_1} & \mathbf{p}_{b_1}^w & \cdots & \mathbf{q}_w^{b_N} & \mathbf{p}_{b_N}^w \end{bmatrix}^T. \tag{76}$$

Therefore, the overall error state of the filter is

$$\delta\mathbf{x}_{all} = \begin{bmatrix} \delta\mathbf{x}_{IMU} & \delta\boldsymbol{\theta}^b & \delta\mathbf{p}_c^b & \delta\boldsymbol{\theta}_1^w & \delta\mathbf{p}_{b_1}^w & \cdots & \delta\boldsymbol{\theta}_N^w & \delta\mathbf{p}_{b_N}^w \end{bmatrix}^T. \tag{77}$$

The measurement residual is a linearized residual about historical IMU pose errors and camera-IMU extrinsic errors. The original reprojection error is manipulated firstly by left nullspace multiplication to marginalize out the feature position, and secondly by applying QR decomposition to decrease the residual dimensions without information loss [1]. Furthermore, only residuals passing through the Mahalanobis gating test would be used in measurement updating.

6.2. Automatic Initialization Procedure

An automatic initialization procedure is developed. Firstly, a stationary scene is automatically detected by only using image stream. Secondly, stationary IMU data is used to initialize the system state. The detailed procedure is described in Algorithm 1.

The algorithm identifies a stationary scene by continuously detecting almost no motion of tracked features. Then, static gyro data is used to initialize gyro bias. Rotation matrix $\hat{\mathbf{R}}_w^b$ is computed by aligning gravity in frame b, which is the mean static accelerometer data, with gravity in frame w. This initialization procedure is a rough one since accelerometer bias has not been eliminated, but its uncertainty can be modeled by the initial covariance matrix of the filter state.

Algorithm 1 Automatic initialization procedure

1. Detect stationary scene

$Counter = 0$
for each image **do**
  $\mathbf{pix}_{curr} = TrackFeatures()$
  if $0 == Counter$ **then**
    $Counter++$
    $\mathbf{pix}_{prev} = \mathbf{pix}_{curr}$
    continue
  end if
  $\mathbf{diff} = \mathbf{pix}_{curr} - \mathbf{pix}_{prev}$
  if $max(\mathbf{diff})$ is small enough **then**
    $Counter++$
  else
    $Counter = 0$
    $\mathbf{pix}_{prev} = \mathbf{pix}_{curr}$
    continue
  end if
  if *Counter* is big enough **then**
    break
  end if
end for

2. Initialize system state

Save stationary acc data in $\mathbf{arry}_{acc}$
Save stationary gyro data in $\mathbf{arry}_{gyro}$
$\mathbf{g}_b = mean(\mathbf{arry}_{acc})$
$\mathbf{g}_w = [0, 0, -9.8]^T$
$\hat{\mathbf{R}}_w^b = FromTwoVectors(\mathbf{g}_b, -\mathbf{g}_w)$
$\hat{\mathbf{b}}_g = mean(\mathbf{arry}_{gyro})$
$\hat{\mathbf{v}}^w = \mathbf{0}; \hat{\mathbf{p}}^w = \mathbf{0}; \hat{\mathbf{b}}_a = \mathbf{0};$

6.3. Refined Feature Triangulation Mechanism

In Mourikis and Roumeliotis's work [1], features are triangulated only if they are no longer being tracked; however, we found that this mechanism does not perform well, especially when using cheap IMUs. To conduct frequent and effective measurement updates, which is crucial to correct biased IMU propagation, a maximum feature tracking length is set. This means each feature would be triangulated when it has been tracked for a certain number of frames, even if it is still being tracked. In the latter situation, the current observation would not be used in triangulation.

Generally, features that failed in triangulation would be discarded directly. While the proposed mechanism is that if a feature fails in triangulation while it is still being tracked, it will have another chance to triangulate when the next image is coming. This mechanism improves the performance when the camera is moving slowly, where features in adjacent images exhibit a small parallax that would easily result in triangulation failure.

7. Experimental Results

The public dataset EuRoC [33] was used to evaluate the performance of the proposed VIO. It includes 11 sequences that were collected by a UAV in three different scenes. One is a machine hall and the other two are rooms equipped with motion capture systems and different manual layout arrangements. The extrinsic and intrinsic parameters of sensors are carefully calibrated, and ground truths of UAV poses are provided. It is one of the widely used benchmarks for evaluating algorithms of different configurations, including

monocular-visual, stereo-visual and monocular/stereo-visual-inertial setups. All of the experiments below were performed on an Ubuntu 16.04 virtual machine powered by MacBook Pro Mid 2015 assigned with two core and 8 GB RAM. Our implementation is a real-time algorithm based on ROS nodelet [41].

The estimated trajectories and corresponding ground truths are shown in Figure 5. Estimated trajectories are aligned with ground truths by a 6-DOF $Sim\,(3)$ transformation without adjusting the scale [42].

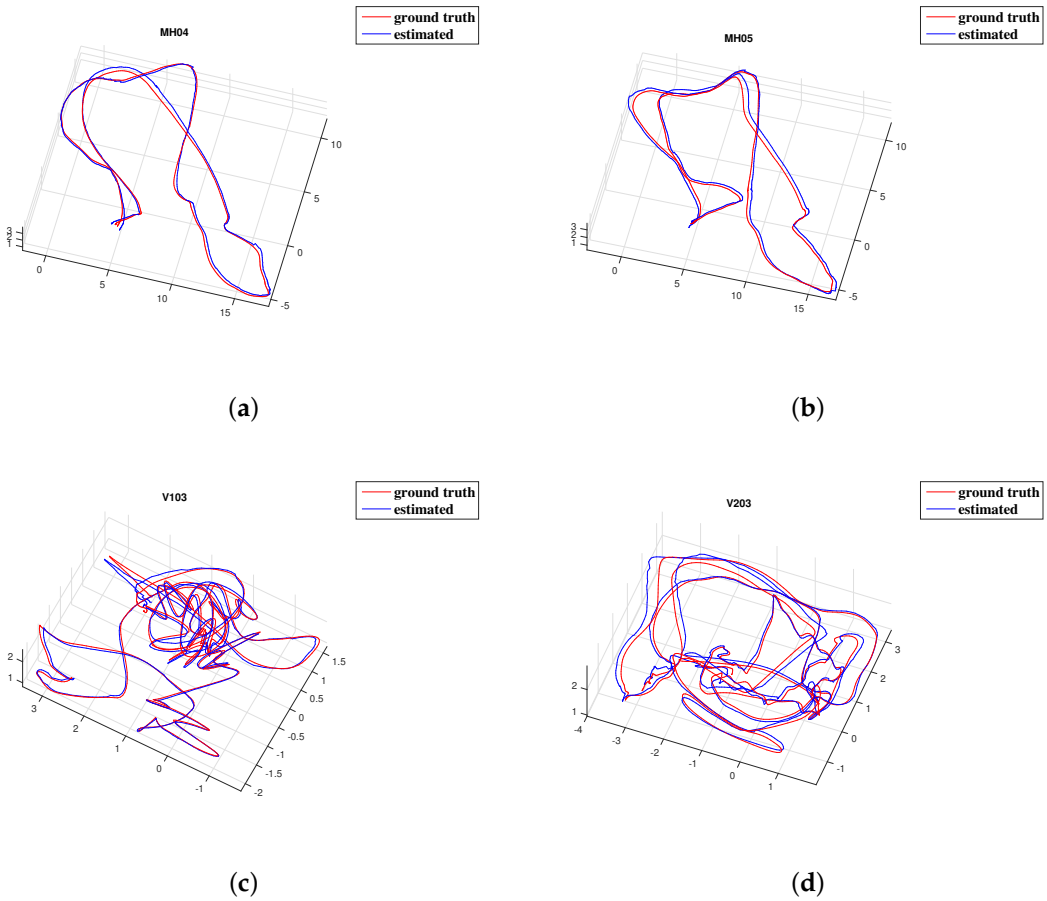

Figure 5. Results of 4 EuRoC sequences classified as "difficult". Estimated trajectories are aligned with ground truths by a 6-DOF $Sim\,(3)$ transformation (without scale). (**a**) MH_04_difficult. (**b**) MH_05_difficult. (**c**) V1_03_difficult. (**d**) V2_03_difficult.

7.1. Front-End Improvement

For implementations of front-ends with and without ORB descriptor assistance, we run each EuRoC sequence 50 times. A boxplot summary is shown in Figure 6. The corresponding means and standard deviations are listed in Table 1.

After adding ORB descriptor assistance, the estimator performs better in most sequences, since boxes became narrow and their position lower in Figure 6. The statistics in Table 1 give a numerical display of the results. Obvious improvement can be observed in seven sequences. In the other four sequences, performance are similar with or without ORB descriptor assistance. This may be due to the small quantity of outliers of optical flow tracking in these sequences.

We also analyzed the processing time of the proposed ORB descriptor-assisted outlier elimination procedure. The maximum feature number is set as 150. The results are listed in Table 2.

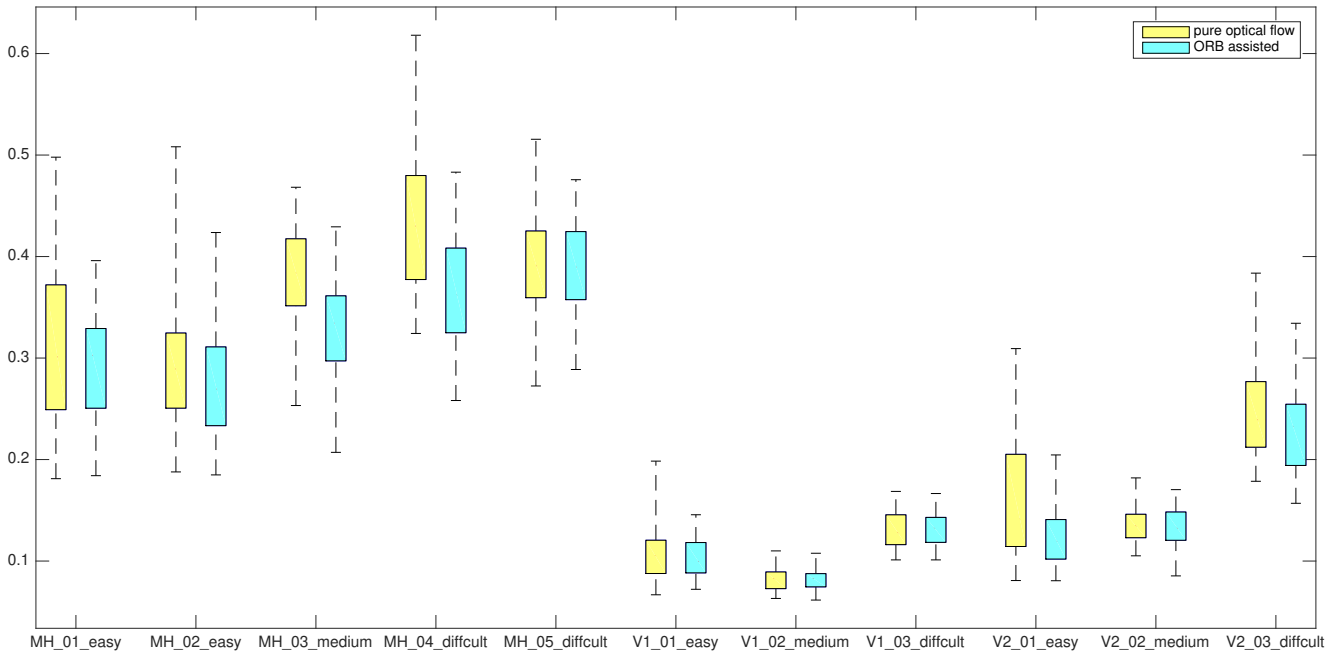

Figure 6. Boxplot summary of experimental results in terms of translation root-mean-square errors (RMSEs) of estimated trajectories. As can be seen, with ORB descriptor assistance the estimation is generally of higher precision, reflected in the lower position and narrower height of the corresponding box's range for most sequences.

Table 1. Mean and standard deviation of RMSEs in Figure 6. For each sequence, the one with an obviously better performance is highlighted.

Sequence	**MH_01**		**MH_02**		**MH_03**		**MH_04**		**MH_05**		**V1_01**		**V1_02**		**V1_03**		**V2_01**		**V2_02**		**V2_03**	
	mean	**std**	**mean**	**std**	**mean**	**std**	**mean**	**std**	**mean**	**std**	**mean**	**std**	**mean**	**std**	**mean**	**std**	**mean**	**std**	**mean**	**std**	**mean**	**std**
pure optical flow	0.309	0.076	0.297	0.065	0.381	0.050	0.435	0.071	0.393	0.051	0.108	0.026	0.082	0.012	0.130	0.018	0.162	0.057	0.137	0.019	0.248	0.047
ORB assisted	**0.294**	**0.055**	**0.273**	**0.056**	**0.330**	**0.048**	**0.366**	**0.058**	0.391	0.046	**0.104**	**0.018**	0.082	0.010	0.131	0.017	**0.127**	**0.030**	0.134	0.019	**0.231**	**0.039**

As shown in Table 2, the proposed ORB descriptor-assisted outlier elimination procedure introduces little computation. The processing time varies among sequences, mostly due to the motion speed. Sequences with aggressive motion tend to take less processing time than those with slow motion since fewer features are tracked in the former case, and fewer ORB descriptor distances need to be calculated.

Table 2. Mean of the processing time (ms) of the proposed ORB descriptor-assisted outlier elimination procedure for every image.

Sequence	MH_01	MH_02	MH_03	MH_04	MH_05	V1_01	V1_02	V1_03	V2_01	V2_02	V2_03
process time	1.3942	1.6480	1.3373	1.3983	1.0870	1.3297	1.0410	0.9574	1.2506	1.0525	0.7465

7.2. Comparison with MSCKF-MONO

We compare our proposed monocular MSCKF with the open-source monocular MSCKF implementation MSCKF-MONO [23]. MSCKF-MONO has a visual front-end based on optical flow and utilizes first-order approximation state transition equations in filtering. It also applies observability-constrained Kalman filter (OC-KF) [21] to fix the observability problem, which would fix the wrong observability properties and improve filter performance. Note that ours does not apply any similar techniques.

In our experiment, we removed the coarse initialization and forbid the reset module in MSCKF-MONO because for some reason, MSCKF-MONO did not work properly on nearly half of sequences under the original coarse initialization, and reset does not help if there is no stop during running. The initial state was assigned by noisy ground truth for both our algorithm and MSCKF-MONO in this experiment. To make a fair comparison, we tried to run with same setup for common parameters in both algorithms, such as noise densities for sensors measurement, sliding window size, and maximum or minimum track lengths for features. However, MSCKF-MONO barely worked in any sequences under a similar setup as ours. This is mainly due to the different state transition model and visual front-end implementations. As we explored further and could not find a setup which generally performed better than the original setup for MSCKF-MONO, we left the original parameters unaltered. The comparison results are listed in Table 3.

Table 3. Comparison results for proposed algorithm and MSCKF-MONO using the EuRoC dataset. The means of positioning RMSEs (m) of 10 runs for both algorithms are calculated.

	MH_01	MH_02	MH_03	MH_04	MH_05	V1_01	V1_02	V1_03	V2_01	V2_02	V2_03
MSCKF-MONO	1.015	0.534	0.427	2.102	0.968	0.169	0.275	1.551	0.281	0.341	×
Proposed	0.299	0.280	0.342	0.350	0.384	0.096	0.078	0.132	0.121	0.137	0.224

The results show that the proposed monocular MSCKF is far more accurate than MSCKF-MONO. We claimed that this is due to a more accurate state transition model and a robust visual front-end.

7.3. Comparison with the State-Of-The-Art

The results of proposed VIO algorithm are compared with several state-of-the-art open-source monocular VIOs using the EuRoC dataset, including OKVIS [6], ROVIO [5], and VINS-MONO [2]. To make a fair comparison between pure VIOs, we turned off the closure detection in VINS-MONO. The proposed VIO automatically selects stationary IMU data to initialize the rotation and gyro bias at the beginning of every sequence, while other states are initialized as zeros. In addition, a unique parameter configuration is applied in all sequences. Results are listed in Table 4.

Table 4. Results of proposed and state-of-the-art VIOs using EuRoC dataset. Ten runs on each sequence and the means of positioning RMSEs (m) are calculated.

	MH_01	MH_02	MH_03	MH_04	MH_05	V1_01	V1_02	V1_03	V2_01	V2_02	V2_03
VINS-MONO	**0.159**	**0.182**	**0.199**	**0.350**	**0.313**	0.090	0.110	0.188	**0.089**	0.163	0.305
ROVIO	0.250	0.653	0.449	1.007	1.448	0.159	0.198	0.172	0.299	0.642	**0.190**
OKVIS	0.376	0.378	0.277	0.323	0.451	**0.087**	0.157	0.224	0.132	0.185	0.305
Proposed	0.289	0.258	0.331	0.394	0.423	0.117	**0.089**	**0.134**	0.097	**0.140**	0.211

As shown above, the proposed VIO algorithm is comparable in accuracy to the state-of-the-art. Notice that VINS-MONO generally performs best out of all four algorithms, and the proposed algorithm has a similar performance in vicon rooms, which is due to good feature triangulation results in a limited area. In addition, the proposed algorithm and ROVIO perform better in V1_03 and V2_03 than others. There are aggressive motions in these two sequences that might result in tracking failure in the front-end; the proposed algorithm and ROVIO are filter-based methods that can utilize IMU measurements to propagate for a short period in this situation, while VINS-MONO and OKVIS sometimes fail and have to lean on relocalization in this circumstance. Notice that the machine hall is a relatively large-scale scenario [33], where triangulations in the proposed method mostly deal with points of large depth. This results in a relatively downgraded performance of the proposed method in the machine hall, even in sequences with mild motions.

7.4. Processing Time

As mentioned by Delmerico and Scaramuzza [17], the better performance of VINS-MONO is a trade-off requiring more computer resources than others. In contrast, the proposed method has a similar architecture to MSCKF-MONO, which is a light-weight solution. The average processing time of the visual front-end and EKF/optimization back-end of our implementation and the state-of-the-art are listed in Table 5.

The results show that, the proposed method has higher processing speed than the listed optimization-based methods. ROVIO is the fastest solution among all listed solutions, but as shown in Table 4, its precision is generally the worst. In proposed method, the visual front-end can process images at about 60 Hz. Notice that V2_03 is a little bit slower than others, because aggressive motions in this sequence result in a short feature tracking length, and thus, the front-end will take more time to extract new features. The EKF-based back-end run at more than 160Hz and the difference between each sequence is due to the difference in the number of features used in measurement updating. As can be concluded from Tables 4 and 5, the proposed method is a VIO solution which has comparable precision and generally required less computation resources than the state-of-the-art.

Table 5. Average processing time (ms) and rate (Hz) of visual front-end and EKF/optimization back-end of our implementation and the state-of-the-art using the EuRoC dataset.

	Sequence	**MH_01**		**MH_02**		**MH_03**		**MH_04**		**MH_05**		**V1_01**		**V1_02**		**V1_03**		**V2_01**		**V2_02**		**V2_03**	
		Time	**Rate**	**Time**	**Rate**	**Time**	**Rate**	**Time**	**Rate**	**Time**	**Rate**	**Time**	**Rate**	**Time**	**Rate**	**Time**	**Rate**	**Time**	**Rate**	**Time**	**Rate**	**Time**	**Rate**
VINS-MONO	front-end	18.0	55	18.3	55	18.6	54	19.3	52	21.3	47	20.2	49	21.4	47	23.2	43	22.3	45	23.8	42	30.6	33
	back-end	50.2	20	50.9	20	50.1	20	50.1	20	53.0	19	53.1	19	45.9	22	37.9	26	54.4	18	48.3	21	33.4	30
ROVIO	front-end	2.0	505	1.9	526	2.0	497	2.1	476	2.0	490	1.9	538	2.0	508	2.1	481	2.0	503	2.0	510	2.0	478
	back-end	15.9	63	15.9	63	15.9	63	15.9	63	15.7	63	15.9	63	15.9	63	15.9	63	15.9	63	15.9	63	15.9	63
OKVIS	front-end	46.7	21	45.3	22	47.4	21	40.9	24	41.4	24	38.5	26	38.8	26	31.3	32	38.8	26	37.3	27	31.4	32
	back-end	39.8	25	39.4	25	39.9	25	32.1	31	33.1	30	30.6	33	25.5	39	19.2	52	29.6	34	27.9	36	18.0	56
Proposed	front-end	16.2	62	16.5	61	15.9	63	16.1	62	15.7	64	15.7	64	15.3	65	16.4	61	15.8	63	15.9	63	17.3	58
	back-end	5.5	182	5.9	169	6.1	164	5.5	181	6.0	166	5.7	174	5.4	185	4.9	203	5.7	176	5.6	178	4.6	218

8. Conclusions

In this paper, we first deduced a highly closed-form IMU error state transition equation from scratch. By using Hamilton's notation of quaternion, we tried to eliminate notation ambiguity. We then managed to solve the integration terms left behind in the transition equation by introducing a two-sample fitting method to approximate the axis-angle, resulting in a fully linear closed-form formulation that is unbiased up to the fitting resolution. This formulation also has potential to incorporate IMU intrinsics into the filter state, since it is a linear function of IMU measurements. An automatic initialization procedure is developed and the feature triangulation mechanism is carefully refined. The ORB descriptor distance between Shi-Tomasi corner pairs was analyzed, and we found that there is a statistical difference in descriptor distances between matched and unmatched feature pairs. As outliers are sometimes fatal for filter-based VIOs, this inspired us to propose a visual front-end based on optical flow tracking and additionally, to use ORB descriptors to eliminate outliers. We implement a monocular VIO under the framework of MSCKF with proposed novelties.

Through a comparison between estimation results with and without the proposed outlier eliminating method, we demonstrate its effectiveness. Furthermore, an experiment was done to compare the proposed method with several state-of-the-art VIOs, both in terms of precision and computation. Results show that the proposed VIO is a visual inertial fusion solution with comparable precision to the state-of-the-ar but which demands less computation resources.

Future works include adding a robust initialization procedure adapting to versatile scenes and analyzing the point selection mechanism in detail.

Author Contributions: X.Q. and H.Z. designed the algorithms. X.Q. deduced all the formulas, analyzed the experimental results and drafted the paper. W.F., C.Z., and Y.J. revised the draft.

Funding: This research work is supported by the National Key Research and Development Program of China (Grant No. 2016YFB0502004 and No. 2017YFC0821102).

Acknowledgments: We would like to thank Sun et al. [24] for their released code. X.Q. would also like to thank Xingwei Qu, Huakun Cui, and Shuhang Liao for their inspiring talks.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mourikis, A.I.; Roumeliotis, S.I. A multi-state constraint Kalman filter for vision-aided inertial navigation. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Roma, Italy, 10–14 April 2007; pp. 3565–3572.
2. Qin, T.; Li, P.; Shen, S. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Trans. Robot.* **2018**, *34*, 1004–1020. [CrossRef]
3. von Stumberg, L.; Usenko, V.; Cremers, D. Direct Sparse Visual-Inertial Odometry using Dynamic Marginalization. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 2510–2517.
4. He, Y.; Zhao, J.; Guo, Y.; He, W.; Yuan, K. PL-VIO: Tightly-Coupled Monocular Visual-Inertial Odometry Using Point and Line Features. *Sensors* **2018**, *18*, 1159. [CrossRef] [PubMed]
5. Bloesch, M.; Omari, S.; Hutter, M.; Siegwart, R. Robust visual inertial odometry using a direct EKF-based approach. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–2 October 2015; pp. 298–304.
6. Leutenegger, S.; Lynen, S.; Bosse, M.; Siegwart, R.; Furgale, P. Keyframe-based visual-inertial odometry using nonlinear optimization. *Int. J. Robot. Res.* **2015**, *34*, 314–334. [CrossRef]
7. Kümmerle, R.; Grisetti, G.; Strasdat, H.; Konolige, K.; Burgard, W. g2o: A general framework for graph optimization. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, 9–13 May 2011 ; pp. 3607–3613.
8. Kaess, M.; Johannsson, H.; Roberts, R.; Ila, V.; Leonard, J.J.; Dellaert, F. iSAM2: Incremental smoothing and mapping using the Bayes tree. *Int. J. Robot. Res.* **2012**, *31*, 216–235. [CrossRef]

9. Liu, H.; Chen, M.; Zhang, G.; Bao, H.; Bao, Y. ICE-BA: Incremental, Consistent and Efficient Bundle Adjustment for Visual-Inertial SLAM. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 1974–1982.
10. Gui, J.; Gu, D.; Wang, S.; Hu, H. A review of visual inertial odometry from filtering and optimisation perspectives. *Adv. Robot.* **2015**, *29*, 1289–1301. [CrossRef]
11. Aqel, M.O.; Marhaban, M.H.; Saripan, M.I.; Ismail, N.B. Review of visual odometry: Types, approaches, challenges, and applications. *SpringerPlus* **2016**, *5*, 1897. [CrossRef] [PubMed]
12. Strasdat, H.; Montiel, J.; Davison, A.J. Real-time monocular SLAM: Why filter? In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Anchorage, AK, USA, 3–7 May 2010; pp. 2657–2664.
13. Triggs, B.; McLauchlan, P.F.; Hartley, R.I.; Fitzgibbon, A.W. Bundle adjustment—A modern synthesis. In Proceedings of the 1999 International Workshop on Vision Algorithms, Corfu, Greece, 20–25 September 1999; Springer: Berlin, Germany, 1999; pp. 298–372.
14. Lourakis, M.I.; Argyros, A.A. SBA: A software package for generic sparse bundle adjustment. *ACM Trans. Math. Softw. (TOMS)* **2009**, *36*, 2. [CrossRef]
15. Hsiung, J.; Hsiao, M.; Westman, E.; Valencia, R.; Kaess, M. Information Sparsification in Visual-Inertial Odometry. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018.
16. Agarwal, S.; Mierle, K. Ceres Solver. Available online: http://ceres-solver.org (accessed on 16 August 2018).
17. Delmerico, J.; Scaramuzza, D. A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots. *Memory* **2018**, *10*, 20.
18. Eade, E.; Drummond, T. Scalable monocular SLAM. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), New York, NY, USA, 17–22 June 2006; Volume 1, pp. 469–476.
19. Li, M.; Mourikis, A.I. Improving the accuracy of EKF-based visual-inertial odometry. In Proceedings of the 2012 IEEE International Conference on Robotics and Automation (ICRA), St Paul, MN, USA, 14–18 May 2012; pp. 828–835.
20. Hesch, J.A.; Kottas, D.G.; Bowman, S.L.; Roumeliotis, S.I. *Observability-Constrained Vision-Aided Inertial Navigation*; Technical Report; University of Minnesota, Departmen of Computer Science & Engineering: Minneapolis, MN, USA, 2012; Volume 1, p. 6.
21. Huang, G.P.; Mourikis, A.I.; Roumeliotis, S.I. Observability-based rules for designing consistent EKF SLAM estimators. *Int. J. Robot. Res.* **2010**, *29*, 502–528. [CrossRef]
22. Li, M.; Mourikis, A.I. High-precision, consistent EKF-based visual-inertial odometry. *Int. J. Robot. Res.* **2013**, *32*, 690–711. [CrossRef]
23. Group of Prof. Kostas Daniilidis, R. Msckf-Mono. Available online: https://github.com/daniilidis-group/msckf_mono (accessed on 16 August 2018).
24. Sun, K.; Mohta, K.; Pfrommer, B.; Watterson, M.; Liu, S.; Mulgaonkar, Y.; Taylor, C.J.; Kumar, V. Robust stereo visual inertial odometry for fast autonomous flight. *IEEE Robot. Autom. Lett.* **2018**, *3*, 965–972. [CrossRef]
25. Zheng, X.; Moratto, Z.; Li, M.; Mourikis, A.I. Photometric patch-based visual-inertial odometry. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 3264–3271.
26. Zheng, F.; Tsai, G.; Zhang, Z.; Liu, S.; Chu, C.C.; Hu, H. Trifo-VIO: Robust and Efficient Stereo Visual Inertial Odometry using Points and Lines. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018.
27. Trawny, N.; Roumeliotis, S.I. *Indirect Kalman Filter for 3D Attitude Estimation*; Technical Report; University of Minnesota, Departmen of Computer Science & Engineering: Minneapolis, MN, USA, 2005; Volume 2.
28. Sommer, H.; Gilitschenski, I.; Bloesch, M.; Weiss, S.M.; Siegwart, R.; Nieto, J. Why and How to Avoid the Flipped Quaternion Multiplication. *arXiv* **2018**, arXiv:1801.07478.
29. Yang, N.; Wang, R.; Gao, X.; Cremers, D. Challenges in monocular visual odometry: Photometric calibration, motion bias, and rolling shutter effect. *IEEE Robot. Autom. Lett.* **2018**, *3*, 2878–2885. [CrossRef]
30. Mur-Artal, R.; Tardós, J.D. Visual-inertial monocular SLAM with map reuse. *IEEE Robot. Autom. Lett.* **2017**, *2*, 796–803. [CrossRef]

31. Bloesch, M.; Burri, M.; Omari, S.; Hutter, M.; Siegwart, R. Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback. *Int. J. Robot. Res.* **2017**, *36*, 1053–1072. [CrossRef]
32. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An efficient alternative to SIFT or SURF. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Barcelona, Spain, 6–13 November 2011; pp. 2564–2571.
33. Burri, M.; Nikolic, J.; Gohl, P.; Schneider, T.; Rehder, J.; Omari, S.; Achtelik, M.W.; Siegwart, R. The EuRoC micro aerial vehicle datasets. *Int. J. Robot. Res.* **2016**, *35*, 1157–1163. [CrossRef]
34. Titterton, D.; Weston, J.L.; Weston, J. *Strapdown Inertial Navigation Technology*; IET: Stevenage, UK, 2004; Volume 17.
35. Solà, J. *Quaternion Kinematics for the Error-State Kalman Filter*; Technical Report; Laboratoire dAnalyse et dArchitecture des Systemes-Centre National de la Recherche Scientifique (LAAS-CNRS): Toulouse, France, 2017.
36. Qin, Y. *Inertial Navigation*; Science Press: Berlin, Germany, 2006. (In Chinese)
37. Qin, Y.; Zhang, H.; Wang, S. *Kalman Filtering and Integrated Navigation Principles*, 3rd ed.; Northwestern Polytechnical University Press: Xi'an, China, 2015. (In Chinese)
38. Mur-Artal, Raúl; Tardós, Juan D. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. [CrossRef]
39. Shi, J.; Tomasi, C. *Good Features to Track*; Technical Report; Cornell University: Ithaca, NY, USA, 1993.
40. Bouguet, J.Y. Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel Corp.* **2001**, *5*, 4.
41. Quigley, M.; Conley, K.; Gerkey, B.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A.Y. ROS: An open-source Robot Operating System. In Proceedings of the ICRA Workshop on Open Source Software, Kobe, Japan, 17 May 2009; Volume 3, p. 5.
42. Umeyama, S. Least-squares estimation of transformation parameters between two point patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **1991**, *4*, 376–380. [CrossRef]